

## 3.2 Knowledge discovery workflow automation, automated data preprocessing

# Automatic Method for Data Preprocessing for the GAME Inductive Modelling Method

Miroslav Čepěk, Miloslav Pavlíček, Pavel Kordík, Miroslav Šnorek

*Dept. of Computer Science and Engineering, Karlovo nám. 13, 121 35 Praha 2, Czech Rep.*

cepekml@fel.cvut.cz

**Abstract.** *In this article we want to present a method for automated selection of preprocessing methods based on genetic algorithms. This method is intended as support for data mining expert and ease his work in routine parts like selection of appropriate preprocessing method (e.g. for data normalisation, outlier detection, etc... ). In this paper we will present the first results of our method.*

### Keywords

Automatic Preprocessing, GAME, Genetic Algorithms.

### 1 Introduction

In this article we want to introduce our new method for automatic selection of preprocessing methods. The method is not intended as replacement of standard data preprocessing process but as support for a data miner.

The standard data preparation is very complex and difficult task. It involves many non-trivial decisions and understanding of the data and the business. In data mining applications the preprocessing is usually made by sequences of commands or more likely by a data-flow diagram. In both approaches the data miner have to carefully select and define order of preprocessing methods. Often selected methods are proved not too successful and data miner have to select another methods or change their order. Since this is long work we want to automate this task. The data preparation process can be divided into two parts. The first involves feature extraction and transformation, "high level" data cleaning, etc... This part requires understanding of nature of the problem and is varies from case to case and therefore it is very hard to automate.

But there is also second part of the data preparation problem – it involves missing values imputation, data normalisation, feature selection, discretisation, nominal values coding, etc... . There are few methods for each task to choose among and it is seldom clear which method use and which is the most suitable for the data. More many methods have several parameters which greatly influences results of the method.

For example to treat missing values one can remove instances with missing values, replace missing value with mean value, copy value from another instance or use some sophisticated algorithm like multiple imputation. Each method affects performance of final model. More influence on model of each method depends on data and therefore it is hard to say which method is the best for given problem.

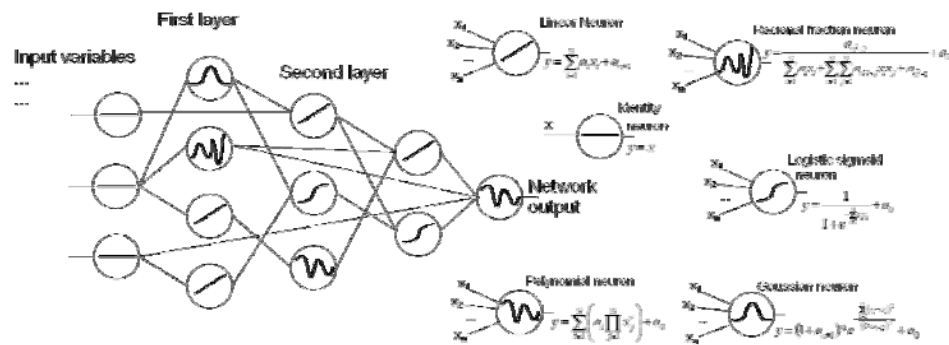
The selection of the best combination of preprocessing methods is therefore difficult and time consuming work. In this paper we want to present genetic algorithm based method for automatic preprocessing methods selection. In this article we will present first results of our method.

This method is intended to support data miner in decision which preprocessing methods select and help him to find optimal setup of selected methods. Naturally it can not replace human expert.

## 2 Theoretical Part

In many applications it is important to find optimal model of unknown system (for example in classification, prediction, approximation, etc.). Such model can be found using two different approaches – deductive and inductive. The GAME artificial neural network (ANN) is based on inductive approach. This means that parameters and also structure of the ANN are constructed during learning from some minimal blocks.

The GAME ANN extends concept of GMDH network [1]. The GMDH allows only one type of minimal block (neurons with one transfer function). On the other hand in GAME ANN there are neurons with many different transfer functions (linear, sigmoid, polynomial, etc...). The GAME have feed-forward structure [2]. Example of GAME ANN architecture is on Figure 1.



**Fig. 1:** Example of the GAME model structure.

The GAME NN is build during the training phase from scratch. Each new layer is constructed following way: first, large number of new neurons is generated, neurons differ in transfer function, number of inputs and neurons in previous layer the new neuron is connected to. The next step is to find optimal setup of internal parameters and best connections to neurons in previous layer (this step will be described later).When this is finished, all neurons are evaluated using separated testing set and the worst neurons are deleted from the layer. Then layer is "frozen" and the algorithm continues with creation of new layer. This is repeated until neuron with satisfactory accuracy is found.

## 3 Theoretical Part

In this part we will describe our method for automatic preprocessing. Our method is based on simple genetic algorithm (GA) in its classical form [3]. The GA consists of three standard steps – selection, cross over and mutation.

In our algorithm each individual in our algorithm represents a sequence of preprocessing methods. The methods to apply are encoded into genome. Each gene encodes one preprocessing method.

Individuals for cross over are selected using tournament selection. Tournament selection works like this - several individuals are selected and their fitness is compared. Two individuals with the highest fitness are selected for cross over. Cross over is standard one point cross over. First in both genomes the cutting points are selected and then corresponding parts of the genome are exchanged.

Mutation is slightly more complex. It may do three different operations. The first is to add or remove preprocessing methods from the genome. Second it may change one method in genome to another. Third, it may alter the configuration of some preprocessing methods.

The fitness function of an individual is computed from accuracy of the GAME model. The methods are applied in order as they are listed in the genome. Then several (three in our case) GAME models are trained using the preprocessed data. Performances of trained models are averaged and this average is taken as the fitness.

## 4 Preprocessing Methods

In this part we want to present types of preprocessing methods we tested in our paper. But first there is an important thing to think about.

### 4.1 Global and Local Preprocessing Methods

There are two types of preprocessing methods – one type usually operates with only one input (attribute) and preserves type of the data. This type we will call local methods. The second type of preprocessing methods changes structure of the data and we will call them global.

The example of the first type of preprocessing methods is method which replaces missing values with predefined constant or normalisation method. The example of second type is PCA transformation.

### 4.2 Implemented Preprocessing Methods

At this time there are several groups of preprocessing methods. The first group contains methods for missing data imputation. This group contains several simpler methods. Among these methods are removing instances with missing values, imputing constant values, mean values and averages of near instances.

The second group contains normalization methods like Z-Score normalisation or Soft-Max normalisation.

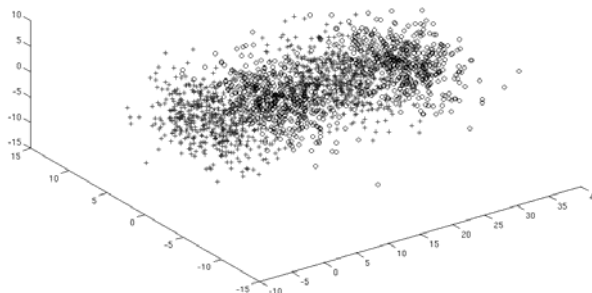
The third group are methods for data reduction. To this group are methods for data reduction. Examples are method for removing random instances, method for removing nearby instances or Principal Component Analysis.

The last group contains rest of methods – discretisation method, white noise generator and some few others methods.

All of these methods can be used in automatic preprocessing.

## 5 Experiments

To test our method we will use one set of artificial data. The set is a classification problem. In this set there are four overlapping clusters from two classes shown on figure 2.



**Fig. 2:** Visualisation of two clusters in artificial data.

The data were obtained as follows: first we generated complete data. These data we divided into training and testing part. Testing part we left as is and will be used to compare results of models. From the training part we removed different portions of data. This way we obtain several different training sets and we will examine which preprocessing methods will be selected and how portion of missing data affects selection of different methods and how successful the models are.

## 6 Results

In this part we will describe our results. First we will discuss our classification experiment.

The results presented in table 1 are obtained following way – the first column presents results for raw data with no preprocessing method applied. The second column show results for simple preprocessing applied. The simple preprocessing means removing all instances with missing values. The third column shows results achieved by automatically preprocessed data.

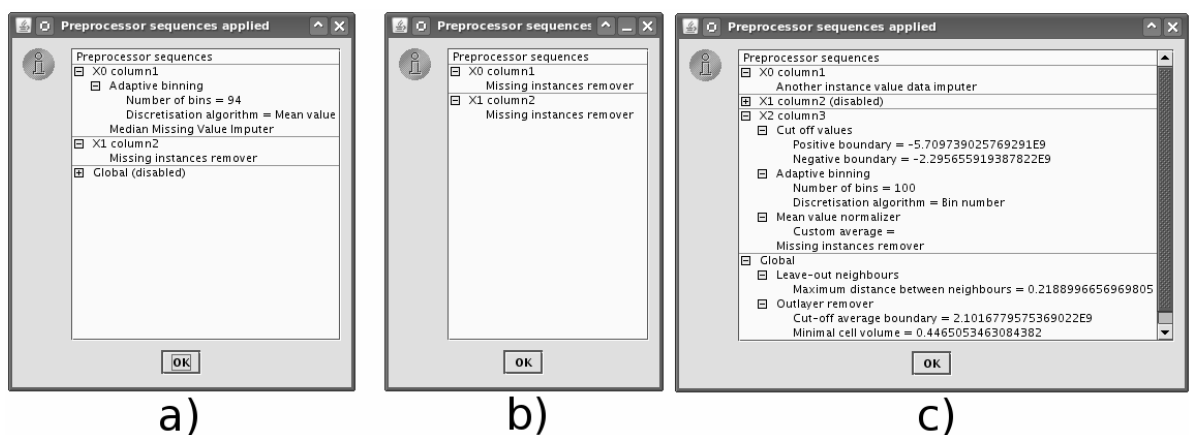
The classification results are summarised in table 1. The first column shows that GAME neural network is very sensitive to missing values and even small amount of missing values can spoil the model.

The second column shows results of data preprocessed by hand. The preprocessing step consists only of removing instances with some value missing. This column shows that even removing instances with missing values improves accuracy.

The third column presents results of automated preprocessing. It shows that that our method for automatic preprocessing is able to find better combinations of preprocessing methods.

	Raw data	Data Preprocessed by hand	Automatically preprocessed data
full data	91.25	91.25	92.5
0.01 missing	55.2	88.79	91.4
0.02 missing	54.2	87.97	91.15
0.05 missing	55	87.5	92.3
0.10 missing	55.26	87.8	99.05
0.20 missing	54.15	80.5	85.7
0.25 missing	54	91.2	92.12
0.50 missing	53.6	89.85	91

**Tab. 1:** Accuracy of models. The left column (Raw data) shows accuracy of models created using raw (not-preprocessed) data. The right column shows accuracy of models created using automatically selected preprocessing



**Fig. 3:** The best individuals for selected amount of missing values. Part a) shows the best chromosome 1% of missing values. Part b) shows individual for 5% of missing values and c) shows 20% of missing values.

As we expected the automatic preprocessing selected mainly methods for treating missing values. Also as we expected all preprocessing sequences contain methods for treating missing data.

Figure 3 illustrates three best individuals for different amount of missing data. The part a) shows the best individual for data with 1% of missing values. This individual treats only two input attributes and therefore resulting models uses only these two input attributes. Missing values are treated using two different methods. Into the first attribute its mean value is imputed and instances containing missing value in the second attribute are removed.

The part b) on figure 3 shows selected methods for 5% missing values. It shows that model also takes only two attributes into account because only two attributes are preprocessed. More it shows that the best chromosome is derived from the predefined individual from the initial population.

The last part c) shows the best individual for 20% of missing values. Models derived from this individual again use only two attributes – the first and the third. This individual removes instances with missing values in the third attribute while it imputes values from neighbouring instances into the first attribute. The rest of preprocessing methods removes nearby instances and slightly reduces number of instances.

We performed two experiments – in the first the initial population was completely generated at random. In this experiment the results were comparable to that of achieved by removing instances with missing values. Generated genomes also contained wide variety of preprocessing methods and their combination.

In the second – which is presented in table 1 – part of initial population was introduced by us and contained method removing instances with missing values. The results in this experiment are slightly better. Also genomes are more uniform and the most of them contains method for removing instances with missing values.

This shows that in the first experiment the population and the number of generations was too small and the genetic algorithm was unable to find better setup. It also shows that it is a good idea to provide the genetic algorithm a reasonable start point.

## 7 Conclusion

In this paper we presented preliminary results of our new method for automatic selection and ordering of preprocessing methods. Our method is based on genetic algorithms and selects preprocessing methods which produce the most accurate GAME model.

Our experiment showed that our automatic preprocessing method is able to find combinations and setup of preprocessing methods which can achieve higher accuracy than simple preprocessing method. The accuracy achieved by automatic preprocessing method is about 4-5% percent higher than accuracy achieved by simple preprocessing method.

The experiment also showed that the automatic preprocessing method performs better when some individuals in the initial population contain reasonable preprocessing methods.

## 8 Acknowledgement

This research is partially supported by the grant "Automated Knowledge Extraction" (KJB201210701) of the Grant Agency of the Academy of Science of the Czech Republic and the research program "Transdisciplinary Research in the Area of Biomedical Engineering II" (MSM6840770012) sponsored by the Ministry of Education, Youth and Sports of the Czech Republic.

## 9 References

- [1] H. Madala and A. Ivakhnenko. Inductive Learning Algorithm for Complex System Modelling. CRC Press, 1994.
- [2] Pavel Kordik. Fully Automated Knowledge Extraction using Group of Adaptive Models Evolution. PhD thesis, Czech Technical University, Prague, March 2007.
- [3] Goldberg D. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, 1989.